I. AMENDMENT

Amendments to the Specification:

Please amend the specification as follows:

Paragraphs beginning at page 1, line 14:

Figure 1 is a diagram of an interlaced video frame 10, which includes an even video field 12 and an odd video field 14. The even field 12 includes the even lines a0, a2, a4 . . . a(n-1) of the frame 10, and the odd field 14 includes the odd lines b1, b3, b5 . . . bn of the frame 10. A video source (not shown) such as a video camera generates the even field 12 at a time t_0 and generates the odd field 14 at a subsequent time t_1, and a video display (not shown in Figure 1) displays the fields 12 and 14 in the same sequence. For example, according to the National Television Standards Committee (NTSC) video standard, which has been in existence for over 50 years, a video source generates and a video display displays one field 12 or 14 every 1/60th of a second, and thus respectively generates and displays one frame 10 every 1/30th of a second. But even though the display displays the fields 12 and 14 at different times, the relatively slow frequency responses of the display and the human eye cause a viewer to perceive that the display is displaying the fields 12 and 14 simultaneously. Thus, the viewer perceives the ~~frame 14 as~~ <u>frame 10 as</u> a single image instead of two sequential partial images.

Many modern video applications, however, generate streams of non-interlaced, *i.e.,* progressive, video frames. For example, most applications of the new High-Definition Television (HDTV) standards such as MPEG (Motion Pictures Experts Group) call for the generation and display of an entire ~~frame 14~~ <u>frame 10</u> approximately every 1/60th of a second. Because such MPEG video sources and displays respectively generate and display all the lines of a progressive frame at one time and not at two sequential times, progressive frames contain little if any motion blurring.

Paragraph beginning at page 9, line 2:

~~Figure 5~~ <u>Figure 4</u> is a block diagram of an image processing circuit 50 according to an embodiment of the invention. The circuit 50 can calculate the values of filler pixels from the original pixels in the original video fields, generate filler fields from the filler pixels, and merge the filler fields and original fields to generate resulting video frames. In one

embodiment, the circuit 50 extends the influence of detected inter-field motion to more filler fields than prior image processing circuits. Thus, the circuit 50 often generates more accurate filler fields and higher-quality resulting frames than prior circuits. In another embodiment, the circuit 50 uses less memory for storing motion information than many prior circuits. In yet another embodiment, the circuit 50 spatially interpolates thin lines and edges more accurately than many prior circuits, and this further increases the accuracy of the filler fields and the visual quality of the resulting frames.

Paragraph beginning at page 10, line 9:

Next, referring to block 72, the processor 56 calculates respective spatial and temporal weighting factors from the motion value and weights the spatially and temporally interpolated pixel values with these respective factors. The processor 56 then combines these weighted pixel values to generate a respective resulting pixel value for each filler pixel in the group. The processor unit 54 stores these resulting filler pixel values in the frame buffer 60, or stores them in the memory 58 until the processor 56 generates the entire filler field.

Paragraph beginning at page 13, line 15:

Referring back to ~~Figure 7~~Figure 4, because the image processing circuit 50 of Figure 5 derives the luminance difference values DY from groups of four original pixels in the same original line, the circuit 50 cannot use the above-described technique to generate raw motion values for the filler pixels at the beginnings and ends of filler lines. For example, the circuit 50 uses four pixels P_{01}, P_{02}, P_{03}, and P_{04} in the same line to generate some of the DY values for the filler block 80. Therefore, P_{01} precedes the block 80 in a horizontal direction, and P_{04} proceeds the block 80 in the horizontal direction. But referring to the filler blocks 82 and 84, because no pixels precede the block 82 and no pixels proceed the block 84, equation (1) is invalid for these blocks. Thus, in one embodiment, the circuit 50 calculates no raw or filtered motion values for the filler blocks 82 and 84 and the other filler blocks that contain the first two or last two filler pixels of respective filler lines. Alternatively, the circuit 50 assigns predetermined filtered motion values to these blocks. For example, in one embodiment, the circuit 50 assigns these blocks a constant FM value or the same FM value as calculated for the adjacent block in the same filler line.

For example, according to the latter approach, the circuit 50 sets the filtered motion value FM_{00} for the block 82 equal to FM_{01}, which the circuit 50 calculates for the adjacent block 80 as described above.

Paragraphs beginning at page 16, line 6:

Referring to Figures 7, 9, and 10, as long as k + 1 is divisible by four, then there are no partial (2 x 1) filler-pixel blocks at the bottom of the complimentary odd filler fields for the even fields E. For example, the last row of filler blocks include respective pixels from the filler lines a(k-2) and a(k), and there are no unpaired filler lines below this. Conversely, if k + 1 is not divisible by four, then there is a row of partial filler blocks that include pixels from only one filler line a(k). In this situation, the image processing circuit 50 of ~~Figure 5~~ Figure 4 can calculate the raw and filtered motion values and the motion-trace values for these partial filler blocks in a number of ways. For example, the circuit 50 can set the raw and filtered motion values and the motion-trace value for a partial filler block equal to the raw and filtered motion values and motion-trace value, respectively, for the full filler block immediately above the partial filler block. For example, referring to Figures 7 and 9, if the filtered-motion-value location $FM_{(k/4)1}$ corresponds to a partial filler block in the filler field that compliments E_0, then the circuit 50 can set $FM_{(k/4)1} = FM_{((k/4)-1)1}$ and $MT_{(k/4)1} = MT_{((k/4)-1)1}$.

Referring to Figure 11, the generation of motion values for filler pixels in ~~even~~ odd filler fields is discussed according to an embodiment of the invention. For example purposes, the generation of motion values is discussed in conjunction with the original odd fields O_0 and O_1 being represented in a Y, C_B, and C_R color space and having been compressed according to the MPEG 4:2:0 format, it being understood that the same principles apply to the other original odd fields of the Figure 6 sequence. The generation of motion values for filler pixels in ~~odd~~ even filler fields is discussed above in conjunction with Figure 7.

Paragraph beginning at page 18, line 14:

Referring to Figure 12, the generation of motion values for filler pixels in ~~odd~~ even filler fields is discussed according to another embodiment of the invention in which the original even fields of Figure 6 are represented in a Y, C_B, and C_R color space and have been compressed and decompressed according to the MPEG 4:2:2 format. Like the

original pixels, the filler pixels are arranged in 2 x 1 blocks of two pixels (see Figure 3A). For example, a block 98 includes filler pixels P_{12} and P_{13}, and a block 100 includes filler pixels P_{32} and P_{33}. Thus, the major difference between Figures 7 and 12 is that in Figure 12, the filler-pixel blocks contain two pixels instead of four pixels. Therefore, DY, DC_R, DC_B, and RM for the block 98 are given by the following equations:

10) $$DY_{ij} = \left|Y_{ij} - Y'_{ij}\right|_{i = 0,2;\, j = 1,2,3,4}$$

11) $$DC_{Ri1} = \left|C_{Ri1} - C'_{Ri1}\right|_{i = 0,2}$$

12) $$DC_{Bi1} = \left|C_{Bi1} - C'_{Bi1}\right|_{i = 0,2}$$

13) $$RM_{01} = Max\left[\frac{1}{8}\sum_{i_{even}=0}^{2}\sum_{j=1}^{4} DY_{ij}, \frac{1}{2}\sum_{i_{even}=0}^{2} DC_{Ri1}, \frac{1}{2}\sum_{i_{even}=0}^{2} DC_{Bi1}\right]$$

FM_{01} is given by equation (5). Furthermore, the calculation of DY is the same as for the 4:2:0 format, and thus equation (10) is the same as equation (5). Furthermore, because the 4:2:2 format calls for one C_R and one C_B value for each 2 x 1 block of original pixels, the calculation of DC_R includes taking the difference between C_{R01}, which corresponds to pixels P_{02} and P_{03} of E_0, and C'_{R01}, which corresponds to P'_{02} and P'_{03} of E_1, and taking the difference between C_{R11}, which corresponds to pixels P_{22} and P_{23} of E_0, and C'_{R11}, which corresponds to pixels P'_{22} and P'_{23} of E_1. A similar analysis applies to DC_B.

Paragraph beginning at page 20, line 1:

In addition, if k +1 is divisible by two, the last filler line a(k) of the filler field that compliments E_0 is not "sandwiched" between two original lines of E_0. Therefore, the circuit 50 calculates DY, DC_R, DC_B, and RM for a last-line filler block such as the block 106 using original pixels in only the last lines a(k-1) and c(k-1), respectively, of the original fields E_0 and E_1. For example, the circuit 50 calculates the difference and raw motion values for the pixel block 106 according to the following equations:

18) $DY_{(k-1)j} = \left| Y_{(k-1)j} - Y'_{(k-1)j} \right|_{j = 1,2,3,4}$

19) $DC_{R(k-1)1} = \left| C_{R(k-1)1} - C'_{R(k-1)1} \right|$

20) $DC_{B(k-1)1} = \left| C_{B(k-1)1} - C'_{B(k-1)1} \right|$

21) $RM_{(k-1)1} = Max\left[\frac{1}{4}\sum_{j=1}^{4} DY_{(k-1)j}, DC_{R(k-1)1}, DC_{B(k-1)1} \right]$

Thus, for example DY for the block 106 is calculated using the luminance values for the pixels $P_{(k-1)1}$, $P_{(k-1)2}$, $P_{(k-1)3}$, and $P_{(k-1)4}$ from E_0 and $P'_{(k-1)0}$, $P'_{(k-1)1}$, $P'_{(k-1)2}$, $P'_{(k-1)3}$, and $P'_{(k-1)4}$ from E_1. Likewise, DC_R and DC_B are calculated from the C_R and C_B values, respectively, for the 2 x 1 blocks of original pixels that include $P_{(k-1)2}$ and $P_{(k-1)3}$ from E_0 and $P'_{(k-1)2}$ and $P'_{(k-1)3}$ from E_1. The circuit 50 uses equation (5) to calculate the filtered motion values ~~including filter equations 5 are used to calculated the filtered motion values~~.

Paragraph beginning at page 22, line 9:

Referring to Figure 15, the generation of motion values for filler pixels in ~~even~~ odd filler fields is discussed for the original odd fields of Figure 6 being represented in a Y, C_B, and C_R color space and having been compressed and decompressed according to the MPEG 4:2:2 format. The calculation of the difference values DY, DC_R, and DC_B and the raw and filtered motion values RM and FM for the even filler fields are similar to the respective DY, DC_R, DC_B, RM, and FM calculations for the ~~odd~~ even filler fields as described above in conjunction with Figure 12. For example, DY, DC_R, DC_B, and RM for the block 112 are given by the following equations:

22) $DY_{ij} = \left| Y_{ij} - Y'_{ij} \right|_{i = 1,3;\ j = 1,2,3,4}$

23) $DC_{Ri1} = |C_{Ri1} - C'_{Ri1}|_{i=1,3}$

24) $DC_{Bi1} = |C_{Bi1} - C'_{Bi1}|_{i=1,3}$

25) $RM_{11} = Max\left[\frac{1}{8} \sum_{i_{odd}=1}^{3} \sum_{j=1}^{4} DY_{ij}, \frac{1}{2} \sum_{i_{odd}=1}^{3} DC_{Ri1}, \frac{1}{2} \sum_{i_{odd}=1}^{3} DC_{Bi1} \right]$

Paragraph beginning at page 24, line 3:

Referring to ~~Figure 5~~ Figure 4 and block 112 of Figure 16, the processor 56 loads 0 into all storage locations of the buffers 90 and 92.

Paragraph beginning at page 24, line 12:

Conversely, referring to blocks 116 and 122, if FM_{kx} is less than the current contents of the ~~location~~ k, x location of the motion-value buffer 90, then the processor 56 analyzes the contents of the k, x, location of the trace buffer 92. If the contents equals 0, then, referring to block 124, the processor 56 loads 0 into the k, x location of the motion-value buffer 90 to indicate that there is no motion associated with the respective filler-pixel block of the current filler field. Conversely, referring to block 126, if the contents of the k, x location of the trace buffer 92 does not equal 0, then the processor 56 decrements the contents by a predetermined value D. In one embodiment, D = 1.

Paragraph beginning at page 28, line 11 (Equation 32):

Next, the processor 56 calculates the final values P_f of the filler pixels according to the following equations:

31) $\alpha = Max(FM_{(i-1)j}, FM_{ij}, FM_{(i+1)j})$

32) $P_f = \frac{1}{15}(\alpha P_s + (\underline{15}[[1]] - \alpha) P_t)$

Paragraph beginning at page 28, line 18:

Specifically, the processor 56 calculates P_f equal to the sum of α-weighted P_s and $(1 - \alpha)$ weighted P_t. α equals the maximum of the FM value of the filler pixel for which P_f is being calculated, and the FM values of the filler-pixel blocks above and below the filler pixel. For example, referring to Figures 7 and 9, α for the filler pixel P_{33} is the maximum of FM_{01} and FM_{11} (there is no filler pixel block above the block 80, and thus no FM location of the buffer 90 above FM_{01}.) Taking the maximum of the three closest FM values in a vertical direction ensures that the spatially interpolated value P_s is given the greatest weight where there is detected motion. Furthermore, as stated above in conjunction with Figure 8, the maximum value of FM, and thus the maximum value of α, is 15. Therefore, the right ~~left~~ side of equation (32) is divided by 15 for normalization.